Japanese Kokai Patent Application No. Hei 6[1994]-252897

Job No.: 228-126455
Ref.: JP 6-252897/PU030094_US/JMF(JILL)/ORDER NO. ART90
Translated from Japanese by the McElroy Translation Company
800-531-9977 customerservice@mcelroytranslation.com

(19) JAPANESE PATENT OFFICE (JP)

(12) KOKAI TOKUHYO PATENT GAZETTE (A)

(11) PATENT APPLICATION PUBLICATION
No. Hei 6[1994]-252897
(43) Publication Date: September 9, 1994

(51) Int. Cl.[5]:	Identification Codes:	Sequence Nos. for Office Use	FI	Technical Disclosure Section
H 04 L 1/16		4101-5K		
G 06 F 13/00	351 E	7368-5B		
H 04 L 12/18				
		8732-5K	H 04 L 11/18	

Examination Request: Not filed

No. of Claims: 22 (Total of 29 pages; FD)

(21) Filing No.: Hei 5[1993]-61203
(22) Filing Date: February 26, 1993

(71) Applicant: 000155469
Nomura Research Institute, Ltd.
1-10-1 Nihonbashi, Chuo-ku, Tokyo

(72) Inventor: Shinji Ichikawa
Nomura Research Institute, Ltd.
Daini Yamagata Biru,
Nihonbashi Software Center
6-7, Nihonbashi Koamicho
Chuo-ku, Tokyo

(72) Inventor: Tatsuya Watabiki
Nomura Research Institute, Ltd.
Yokohama Research Center
134 Kobecho, Hodogaya-ku
Yokohama-shi, Kanagawa-ken

(74) Agent: Kenji Ushihisa, patent attorney

Continued on last page

(54) Title: Multiple Address File Transfer Method and System

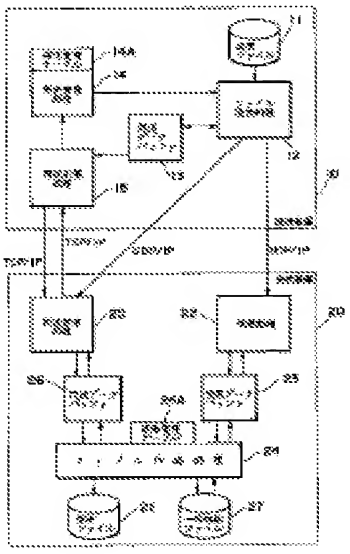
(57) Abstract

Objective

To reliably manage the start and finish of a file transfer and to avoid data loss during [the transfer] when a file is distributed to multiple addresses.

Constitution

A transmitting device 10 reads from a transmission file 11 a file to be transferred, performs data compression, and then divides the same into multiple blocks. The initial text is assigned the serial number 0, and includes the last serial number for the last text. The successive text is assigned a sequentially incremented serial number, and includes block data. The last serial number is appended to the last text. These texts are multiple addresses distributed at fixed time intervals from a file transmission process 12 by means of UDP/IP. At receiving devices 20, a file creation process 24 performs a serial number check, and upon discovery that a number is missing, it transmits a transmission request to a retransmission request process 25. For a first transmission request, file transmission process 12 redistributes [the text] by means of UDP/IP, and for a second retransmission request, a retransmission reply process 15



Key: 10 Transmitting device
11 Transmission file

retransmits to the requesting retransmission request process 25 by means of TCP/IP.

12	File transmission process
13	Retransmission data buffer
14	Retransmission management process
14A	Retransmission table
15	Retransmission reply process
20	Receiving device
21	Reception file
22	Reception process
23	Reception data buffer
24	File creation process
24A	Serial number management table
25	Retransmission request process
26	Retransmission data buffer
27	Temporary storage file

[There are no amendments to this patent.]

Claims

1. A multiple address file transfer method wherein a transmitting station

divides a file to be transferred into multiple blocks,

uses a communication protocol suitable for multiple address distribution to multiple-distribute to multiple receiving stations a starting text containing an initial serial number and a last serial number that should be included with the final electronic text, successive text that includes file data blocks that are divided and incremented sequentially beginning with the initial serial number, and that is transmitted after the initial electronic text, and an end text that contains at least the last serial number and that is transmitted last,

and when there is a retransmission request from a receiving station specifying a serial number, retransmits the text having the specified serial number,

and each receiving station

receives the text distributed from the transmitting station,

checks the serial numbers included in the received text and, upon determination that a number is missing, transmits a retransmission request specifying the missing serial number,

and stores in a file the text that has undergone the serial number check.

2. The multiple address file transfer method recorded in Claim 1 wherein,

when the serial numbers included in the received text are checked and it is determined that a number is missing,

a receiving station transmits a first retransmission request specifying the missing serial number,

and when text that includes the specified serial number still has not been received after a prescribed period of time has elapsed after transmission of the first retransmission request, transmits a second retransmission request,

and when there is a first retransmission request from any of the receiving stations,

the transmitting station multiple-address redistributes the text with the specified serial number using a communication protocol suitable for multiple-address distribution,

and, when there is a second retransmission, retransmits the requested text to the receiving station that transmitted the second retransmission request.

3. The multiple address file transfer method recorded in Claim 1 wherein the transmitting station transmits text periodically at intervals of a first fixed time period,

and the receiving stations check to see if text has been received during a second fixed time period that is longer than the aforementioned first fixed time period, and when there is none, [the receiving stations] determine that an obstruction has been detected and transmit an obstruction notification to the transmitting station.

4. The multiple address file transfer method recorded in Claim 3 wherein,

when an obstruction notification has been received, the transmitting station distributes dummy text at intervals of the aforementioned first fixed time period,

and when the dummy text is received by the receiving station that transmitted the obstruction notification, [the receiving station] transmits a recovery notification to the transmitting station.

5. The multiple address file transfer method recorded in Claim 4 wherein,

when a recovery notification has been transmitted within a prescribed time period from all of the receiving stations that transmitted an obstruction notification, the transmitting station restarts the distribution of the aforementioned initial, successive, or final electronic text.

6. The multiple address file transfer method recorded in Claim 4 wherein,

if a receiving station that transmitted an obstruction notification has not transmitted a recovery notification within a prescribed time period, the transmitting station notifies that receiving station of a mandatory stop.

7. The multiple address file transfer method recorded in Claim 6 wherein the transmitting station stores in memory a receiving station that has been notified of a mandatory stop.

8. The multiple address file transfer method recorded in Claim 6 wherein,

if there is a receiving station that does not transmit an obstruction notification or if there is a receiving station that has transmitted a recovery notification within the aforementioned prescribed time period, the transmitting station restarts the distribution of the aforementioned initial, successive, or final electronic text.

9. The multiple address file transfer method recorded in Claim 1 wherein

the transmitting station applies a data compression process to a file to be transferred and then divides the same into multiple blocks,

and the receiving stations apply an expansion process to the received data blocks after all of the text has been received, and store the file obtained by means of the expansion process in a reception file.

10. The multiple address file transfer method recorded in Claim 1 wherein the transmitting station includes in the aforementioned starting text an execution shell script that indicates a process to be executed by the receiving stations.

11. A multiple address file transfer system comprised of a transmission device that possesses files to be transferred, and multiple receiving devices capable of communicating with this transmitting device,

the transmitting device being equipped with

a distribution means that divides a file to be transferred into multiple blocks and uses a communication protocol suitable for multiple address distribution to multiple-address distribute to multiple receiving stations a starting text containing an initial serial number and a last serial number that should be included with the final electronic text, successive text that includes file data blocks that are divided and incremented sequentially beginning with the initial serial number, and that is transmitted after the initial electronic text, and an end text that contains at least the last serial number and that is transmitted last; and a retransmission means that, when there is a retransmission request from a receiving station specifying a serial number, retransmits the text having the specified serial number,

and each receiving device being equipped with

a receiving means that receives text distributed from the transmitting station,

a retransmission request means that checks the serial numbers included in the received text and, upon determination that a number is missing, transmits a retransmission request specifying the missing serial number, and a means that stores in a file the text that has undergone the serial number check.

12. The multiple address file transfer system recorded in Claim 11 wherein the aforementioned retransmission request means for the receiving device is comprised of

a first retransmission request means that checks the serial numbers included in the received text and, upon determination that a number is missing, transmits a first retransmission request specifying the missing serial number,

and a second retransmission request means that, when text that includes the specified serial number still has not been received after a prescribed period of time has elapsed after transmission of the first retransmission request, transmits a second retransmission request,

and the aforementioned retransmission means for the transmitting device is comprised of

a first retransmission means that, when there is a first retransmission request from any of the receiving stations, multiple-address redistributes the text with the specified serial number using a communication protocol suitable for multiple-address distribution,

and a second retransmission means that, when there is a second retransmission request, retransmits the requested text to the receiving station that transmitted the second retransmission request.

13. The multiple address file transfer system recorded in Claim 11 wherein

the aforementioned distribution means of the transmitting device transmits text periodically at intervals of a first fixed time period,

and the receiving devices are equipped with an obstruction detection means that checks to see if text has been received during a second fixed time period that is longer than the aforementioned first fixed time period, and when there is none, [the detection means] determines that an obstruction has been detected and transmits an obstruction notification to the transmitting station.

14. The multiple address file transfer system recorded in Claim 13 wherein

the aforementioned distribution means of the transmission device distributes dummy text at intervals of the aforementioned first fixed time period when an obstruction notification has been received,

and the receiving devices are provided with a recovery notification means that, when the dummy text is received by the receiving means after the aforementioned obstruction detection means has transmitted an obstruction notification, transmits a recovery notification to the transmitting station.

15. The multiple address file transfer system recorded in Claim 14 wherein

the aforementioned distribution means of the transmitting device restarts the distribution of the aforementioned initial, successive, or end text when a recovery notification has been transmitted within a prescribed time period from all of the receiving stations that transmitted an obstruction notification.

16. The multiple address file transfer system recorded in Claim 14 wherein

the transmitting device is equipped with a means that, if a receiving station that transmitted an obstruction notification has not transmitted a recovery notification within a prescribed time period, notifies that receiving station of a mandatory stop.

17. The multiple address file transfer system recorded in Claim 16 wherein the aforementioned notification means of the transmitting device stores in memory a receiving station that has been notified of a mandatory stop.

18. The multiple address file transfer system recorded in Claim 16 wherein

the aforementioned distribution means of the transmitting device restarts the distribution of the aforementioned initial, successive, or end text if there is a receiving station that does not transmit an obstruction notification or if there is a receiving station that has transmitted a recovery notification within the aforementioned prescribed time period.

19. The multiple address file transfer system recorded in Claim 11 wherein

the aforementioned distribution means of the transmitting device applies a data compression process to a file to be transferred and then divides the same into multiple blocks, and then distributes [said blocks],

and the aforementioned storage means for the receiving stations apply an expansion process to the received data blocks after all of the text has been received, and store the file obtained by means of the expansion process in a reception file.

20. The multiple address file transfer system recorded in Claim 11 wherein the aforementioned distribution means of the transmitting device includes in the aforementioned starting text an execution shell script that indicates a process to be executed by the receiving stations.

21. A transmitting device for the purpose of multiple address file transfer, equipped with

a transmission file that stores file data to be transferred,

a distribution means that reads file data from the aforementioned file, divides this file data into multiple blocks, and uses a communication protocol suitable for multiple address distribution to multiple-address distribute to multiple receiving stations a starting text containing an initial serial number and a last serial number that should be included with the final electronic text, successive text that includes file data blocks that are divided and incremented sequentially beginning with the initial serial number, and that is transmitted after the initial electronic text, and an end text that contains at least the last serial number and that is transmitted last, and a retransmission means that, when there is a retransmission request from a receiving station specifying a serial number, retransmits the text having the specified serial number.

22. The transmitting device for the purpose of multiple address file transfer recorded in Claim 21

wherein the aforementioned retransmission means is comprised of

a first retransmission means that, when a first retransmission request is received, multiple-address redistributes the text with the specified serial number using a communication protocol suitable for multiple-address distribution,

and a second retransmission means that, upon a second retransmission request for text with the same serial number, retransmits the requested text to the receiving station that transmitted the second retransmission request.

Detailed explanation of the invention

[0001]

Technical field

This invention pertains to a multiple address file transfer method wherein data that constitute files (file data) are transferred from a transmitting device to multiple receiving devices using communication lines or radio waves.

[0002]

Prior art

The most critical condition required for a file transfer is that all of the data that constitute the file be correctly transferred; a loss of data cannot be permitted. If a communication protocol (for example, TCP/IP) is used that is suitable for connection-type communication that guarantees the reliability of the data transfer, a reliable file transfer can be achieved.

[0003]

However, a connection-type communication protocol assumes that there is one-to-one communication between the transmitting device and the receiving device; if multiple receiving devices are receiving the transferred file, not only must the file transfer be repeated for each of the received files, but as the number of receiving devices increases, a problem arises in that the load on the transmitting device and the transmission medium increases, and more time is required.

[0004]

Some protocols (for example UPD/IP) are suitable for multiple address distribution wherein data are transmitted at the same time from a transmitting device to multiple receiving devices, but when such a communication protocol is used, a problem arises in that the reliability of the data distribution is insufficient, because whether the data actually have arrived at a receiving device cannot be confirmed.

[0005]

The first item that must be confirmed is whether data have been lost. In addition, if data have been lost, the relevant data must be retransmitted. Furthermore, with a file transfer, the start and end of the file transfer must be confirmed.

[0006]

Next, the management of obstructions is required for the transmitting device, the receiving devices, and the transmission path therebetween. This includes the detection of the obstruction and the response thereto. When an obstruction occurs, data that have been lost as a result thereof must be retransmitted. If the obstruction is resolved, data transmission must be restarted, and if the obstruction is not resolved, the file transfer to the relevant receiving device must be stopped mandatorily.

[0007]

It also is necessary to keep track of the receiving devices to that file transfer has been mandatorily stopped.

[0008]

Summary of the invention

An objective of this invention is to provide a method and a system wherein data that constitute files can be transferred to multiple receiving devices from start to finish reliably and efficiently.

[0009]

Furthermore, an objective of this invention is to provide a method and a system wherein lost data can be retransmitted reliably if there has been a data loss.

[0010]

Furthermore, an objective of this invention is to provide a method and a system wherein an obstruction can be handled reliably if an obstruction occurs.

[0011]

Furthermore, an objective of this invention is to provide a method and a system with which it is possible to manage whether the file transfer to the multiple receiving devices was completed successfully.

[0012]

The multiple file transfer method according to this invention is one wherein a transmitting station divides a file to be transferred into multiple blocks, uses a communication protocol suitable for multiple address distribution to multiple-address distribute to multiple receiving stations a starting text containing an initial serial number and a last serial number that

should be included with the final electronic text, successive text that includes file data blocks that are divided and incremented sequentially beginning with the initial serial number, and that is transmitted after the initial electronic text, and an end text that contains at least the last serial number and that is transmitted last, and when there is a retransmission request from a receiving station specifying a serial number, retransmits the text having the specified serial number.

[0013]

Each receiving station receives the text distributed from the transmitting station, checks the serial numbers included in the received text and, upon determination that a number is missing, transmits a retransmission request specifying the missing serial number, and stores in a file the text that has undergone the serial number check.

[0014]

The multiple file transfer method according to this invention is comprised of a transmitting device that possesses a file to be transferred and multiple receiving devices capable of communicating with this transmitting device.

[0015]

The transmitting device is equipped with a distribution means that divides a file to be transferred into multiple blocks and uses a communication protocol suitable for multiple address distribution to multiple-address distribute to multiple receiving stations a starting text containing an initial serial number and a last serial number that should be included with the final electronic text, successive text that includes file data blocks that are divided and incremented sequentially beginning with the initial serial number, and that is transmitted after the initial electronic text, and an end text that contains at least the last serial number and that is transmitted last; and a retransmission means that, when there is a retransmission request from a receiving station specifying a serial number, retransmits the text having the specified serial number.

[0016]

Each receiving device is equipped with a receiving means that receives text distributed from the transmitting station, a retransmission request means that checks the serial numbers included in the received text and, upon determination that a number is missing, transmits a retransmission request specifying the missing serial number, and a means that stores in a file the text that has undergone the serial number check.

[0017]

By means of this invention, block data that constitute a file are multiple-address distributed from one transmitting station (communication device) to multiple receiving stations (receiving devices), so the file can be transferred to multiple receiving stations efficiently in a short period of time. Furthermore, starting text having a prescribed serial number that is set in advance, successive text having a sequentially incremented serial number, and end text having a last serial number, the notice of which is provided in the initial electronic text, are used to distribute the file as block data, so management of the initial and final [portions of the file transfer] is enabled in addition to management of the serial numbers, and the file can be transferred reliably. Furthermore, even if text is lost, said text is retransmitted in response to a retransmission request, so various errors can be satisfactorily handled.

[0018]

In a preferred embodiment of this invention, when the serial numbers included in the received text are checked and it is determined that a number is missing, a receiving station transmits a first retransmission request specifying the missing serial number, and when text that includes the specified serial number still has not been received after a prescribed period of time has elapsed after transmission of the first retransmission request, [the receiving station] transmits a second retransmission request.

[0019]

In addition, when there is a first retransmission request from any of the receiving stations, the transmitting station multiple-address redistributes the text with the specified serial number using a communication protocol suitable for multiple-address distribution, and when there is a second retransmission, [the transmitting station] retransmits the requested text to the receiving station that transmitted the second retransmission request.

[0020]

For the first retransmission request the missing text is redistributed to all of the receiving stations, so even if the multiple receiving stations are unable to receive identical text, a first retransmission process will suffice, so the process is simple. A highly reliable communication protocol can be used with the second retransmission, so the retransmission is reliable.

[0021]

In another embodiment of this invention the transmitting station transmits text periodically at intervals of a first fixed time period, and the receiving stations check to see if text

has been received during a second fixed time period that is longer than the aforementioned first fixed time period, and when there is none, [the receiving stations] determine that an obstruction has been detected and transmit an obstruction notification to the transmitting station

[0022]

Thus the transmitting can recognize that a transmission path obstruction has occurred.

[0023]

In a preferred embodiment, when an obstruction notification has not been [sic; has been] received, the transmitting station distributes dummy text at intervals of the aforementioned first fixed time period, and when the dummy text is received by the receiving station that transmitted the obstruction notification, [the receiving station] transmits a recovery notification to the transmitting station.

[0024]

By having receiving stations that have not detected an obstruction receive the dummy text, it is possible to know if an obstruction has occurred at any of the receiving stations. Furthermore, by having a receiving station that has detected an obstruction receive the dummy electronic text, recovery from an obstruction is detected, and notice thereof is provided to the transmitting station, so the transmitting station can recognize that there has been a recovery from the obstruction.

[0025]

When a recovery notification has been transmitted within a prescribed time period from all of the receiving stations that transmitted an obstruction notification, the transmitting station restarts the distribution of the aforementioned initial, successive, or final electronic text; therefore, continuation of the file transfer is enabled.

[0026]

If a receiving station that transmitted an obstruction notification has not transmitted a recovery notification within a prescribed time period, the transmitting station notifies that receiving station of a mandatory stop, and stores in memory the receiving station that has been notified of a mandatory stop.

[0027]

Thus it is possible to be aware of and to manage the receiving stations for which the file transfer was completed and the receiving stations for which it could not be completed.

[0028]

With an even more preferable embodiment of this invention, the transmitting station applies a data compression process to a file to be transferred and then divides the same into multiple blocks, and the receiving stations apply an expansion process to the received data blocks after all of the text has been received, and store the file obtained by means of the expansion process in a reception file.

[0029]

Because the data to be transferred are compressed, the transfer time can be reduced and the efficiency can be increased.

[0030]

By having the transmitting station include in the aforementioned starting text an execution shell script that indicates a process to be executed by the receiving stations, it is possible to instruct the receiving stations regarding the process used with the received file.

[0031]

Explanation of application examples

Figure 1 shows the overall configuration of a multiple address file transfer system.

[0032]

One transmitting device 10 and multiple receiving devices 20 are linked by means of a transmission path 8 and, as needed, via a relay device 9 so as to be capable of communicating with each other. Transmitting device 10 and the receiving devices 20 typically are comprised of work stations. Transmission path 8 typically is implemented with a communication line. As the communication line, for example, an Ethernet LAN (Local Area Network) or a WAN (Wide Area Network) can be used. A portion of transmission path 8 can include an existing public line or dedicated line, or one to be added in the future. In this case transmitting device 10 and a receiving device 20 could be located far apart from each other. A communication satellite can be considered for relay device 9. In this case a land-based network (including the aforementioned public line or dedicated line) also is used to supplement the transmission path that is via the communication satellite. In either case, transmission path 8 is capable of transferring data using

both the connection type TCP/IP (Transmission Control Protocol/Internet Protocol) and the connectionless type UDP/IP (User Datagram Protocol/Internet Protocol) (suitable for multiple address distribution) protocols. Transmission device 10 and receiving devices 20 also include computers capable of using these communication protocols.

[0033]

Figure 2 shows the functional configuration of transmitting device 10 and a receiving device 20, as well as the flow of data between these and the internal data flow. The various 'processes' to be explained later are implemented by the operation of a computer in response to a processing program (a process or program module). Each of these processes communicates with the others by inter-program communication.

[0034]

Transmitting device 10 includes a transmission file 11, a file transmission process 12, a retransmission data buffer 13, a retransmission management process 14, and a retransmission reply process 15. A retransmission management table 14A is associated with retransmission management process 14. Transmission file 11 is implemented by means of a recording medium that stores files to be transmitted to receiving devices 20.

[0035]

Receiving device 20 is provided with a reception file 21, a reception process 22, a reception data buffer 23, a file creation process 24, a retransmission request process 25, a retransmission data buffer 26, and a temporary storage file 27. File creation process 24 is provided with a serial number management table 24A. Temporary storage file 27 temporarily stores received file data, after which restored (having undergone an expansion process that will be explained later) final file data are stored in reception file 21. These files 21 and 27 are implemented, for example, by means of a semiconductor memory or a magnetic recording medium.

[0036]

As will be explained later, a file stored in transmission file 11 undergoes compression processing and then is divided into multiple blocks that are then compiled so as to form text with each block, and then distributed. The text is transmitted at a fixed period T1.

[0037]

The multiple address distribution of file data from transmitting device 10 to receiving devices 20 is executed from file transmission process 12 of communication [sic; transmitting] device 10 to reception process 22 of receiving devices 20 using UDP/IP.

[0038]

When an error occurs in the file data transferred from transmitting device 10 to receiving devices 20 by multiple address distribution, a retransmission request is sent from a receiving device 20 to transmitting device 10. In other words, in retransmission request process 25, the receiving device 20 transmits a retransmission request to retransmission reply process 15 of transmitting device 10 using TCP/IP, which guarantees reliability.

[0039]

For the first retransmission request (this will be called 'retransmission request 1'), file transmission process 12 of transmitting device 10 uses UDP/IP to retransmit to retransmission request process 25 of the receiving devices 20 the file data for which a retransmission was requested. Therefore, even if there are duplicate retransmission requests for the same file data from multiple receiving devices 20, they can be handled with one retransmission.

[0040]

For the second retransmission request for the same file data (this will be called 'retransmission request 2'), retransmission reply process 15 of transmitting device 10 uses TCP/IP to ensure reliability, and retransmits, to retransmission request process 25 of the receiving devices 20 that requested the retransmission, the file data for which a retransmission was requested.

[0041]

When a transmission path obstruction occurs, as will be explained later, transmission process 12 of transmitting device 10 distributes dummy text with fixed period T1. This also is performed using UDP/IP.

[0042]

The obstruction notification recovery notification, end notification, and the like transmitted from receiving device 20 to transmitting device 10 are transmitted with respect to retransmission reply process 15 by retransmission request process 25 using TCP/IP.

[0043]

The identification of the source and destination for a multiple address distribution, retransmission request, retransmission, and the various notifications is achieved by using an address for devices (work stations) 10 and 20, and by using port numbers for the various processes 12, 15, 22, and 25.

[0044]

Figure 3 shows the file transfer text format used when transmitting device 10 distributes or retransmits file data to a receiving device 20. Here, the header information according to UDP/IP or TCP/IP has been omitted from the figure.

[0045]

There are four types of electronic text: initial electronic text, successive electronic text, final electronic text, and dummy electronic text. Starting text is used at the beginning of one transmission for one file, and end text is used at the end thereof. Successive text is used essentially to transfer file data in the interval between transmission of the starting text and transmission of the final electronic text. Dummy text is text without data, and is used when an obstruction occurs when the operation of transferring text at the fixed period T1 is underway. By means of this dummy text, a receiving device 20 is able to recognize when an obstruction has been resolved and when an obstruction has occurred with another receiving device.

[0046]

This text is comprised of [the following] fields: a text class, a resource ID, text class details, a file transfer ID, record length, and data (the main body).

[0047]

The system shown in Figure 1 and Figure 2 is used not only for file transfers but for real-time data distribution. The text class indicates, for example, whether it is a file transfer, real-time data distribution, or for another purpose. The resource ID is an identifying code for the data source. For a file transfer, the text class and resource ID are both set on 'FT (data transfer)'.

[0048]

The text class details distinguish whether it is starting text ('ST'), successive text ('MD'), end text ('LS'), or dummy text ('DM').

[0049]

The file transfer ID is an appended identifying code that differs for each file transfer. This is because the same file may be transferred multiple times at different dates and times.

[0050]

The serial number indicates the sequence at which text is distributed; it is incremented by one each time one text is distributed. As will be explained later, this serial number is used effectively to check whether there is an omission in the text received by a receiving device 20. The serial number for the first text is fixed at '0'. The serial numbers for successive texts are allocated sequentially and continuously as positive integers beginning with '1'. The serial number for the end text is the last number that is included in the data (the main body) of the next indicated starting text. It is sufficient if the serial numbers of the starting text and the end text are known in advance, so they are not limited to the above description; any code can be used.

[0051]

The record length indicates the length of the data (main body) included in each text.

[0052]

The content of the data (main body) differs according to the type of text. For the starting text, the data (main body) include the file name of the file being transferred, the size of that file, the last serial number used with the end text, and an execution shell script. The execution shell script indicates to the receiving device that has received the file the content of the process to be executed using that file. With successive text and end text, the data (main body) are file data (that have undergone compression processing and have been divided into blocks) to be transferred.

[0053]

The serial number, record length, and data (main body) fields are not included in dummy text.

[0054]

Figure 4 and Figure 5 show the process flow when a file transfer occurs correctly with no data loss and no obstruction.

[0055]

File transmission process 12 of transmitting device 10 reads from transmission file 11 a file to be transferred (all of the file data), performs data compression, and divides the compressed data into multiple blocks having a length suitable for the file transfer (step 31). In this application example, the data are divided into 100 blocks. Accordingly, the last serial number of the end text is '100'.

[0056]

Next, file transmission process 12 transmits a start notification to retransmission management process 14 (step 32). Having received the start notification, retransmission management process 14 moves from the idle state to the communication state (step 61).

[0057]

File transmission process12 compiles the starting text and distributes this to reception process 22 of receiving devices 20 using UPD/IP (step 33). After delivering the starting text, file transmission process 12 next compiles the successive text and distributes it in the same manner to reception process 22 (steps 33, 34). Finally, the end text is compiled and distributed to reception process 22 (step 35).

[0058]

These texts are distributed at a fixed period of time T1. A serial number that is incremented per transmission of text is, of course, appended to each text.

[0059]

With each transmission of text, file transmission process 12 also transfers the transmitted text to retransmission data buffer 13 and stores it therein. This is in preparation for retransmission requests from the receiving devices 20.

[0060]

When reception process 22 of receiving devices 20 receives the starting text, it store this in reception data buffer 23 and notifies file creation process 24 that text has been received. In addition, it sets a timer for the purpose of detecting obstructions (step 101).

[0061]

As described above, file transmission process 12 sends some sort of text (including dummy text, to be explained later) at a fixed period T1, so if no text has been received during a

prescribed time T2 that is greater than this period T1, then it can be concluded that an obstruction has occurred. The timer for detection of obstructions is set every time any sort of text is received (in the explanation that follows, this point will not be mentioned repeatedly).

[0062]

When notification of the reception of text is received, file creation process 24 stores the serial number included therein in serial number management table 24A, confirms that it is starting text, and opens temporary storage file 27 and reception file 21 (step 121).

[0063]

When successive text is received, reception process 22 writes this to reception data buffer 23 and notifies file creation process 24 that text has been received (step 102).

[0064]

When notification of the reception of text is received, file creation process 24 performs a serial number check. The serial number for the previously received text is stored in serial number management table 24A. If the value of the serial number for the text received at this time is that of the serial number for the previously received text incremented by one, the serial number for the text received at this time is correct. If there is a difference of two or more between the serial number for the text received at this time and the serial number for the previously received text, it is concluded that data are missing for some reason. Furthermore, if the serial number for the text received at this time is identical to the serial number for the text previously received, it means that the text received at this time is a duplicate of the previous text.

[0065]

When the conclusion based on the serial number check is that the serial number for the text received at this time follows the serial number of the text previously received file, creation process 24 writes the serial number for the text received at this time to serial number management table 24A to update the serial number, reads the text received at this time from reception data buffer 23, and writes it to temporary storage file 27 (step 122).

[0066]

Reception process 22 and file creation process 24 respectively repeat the aforementioned steps 102 and 122 each time successive text is received.

[0067]

When end text is received, reception process 22 writes this end text to reception data buffer 23 and transmits a reception notification to file creation process 24 (step 103).

[0068]

File creation process 24 performs a serial number check, and if it is OK, a serial number update is performed and the end text that is read from reception data buffer 23 is written to temporary storage file 27 (step 123).

[0069]

When the end text is received, it means that all of the file data have been collected, so file creation process 24 reads out the reception data stored in temporary storage file 27 and performs a data expansion process. File creation process 24 writes the thus restored file to reception file 21 and closes files 27 and 21. File creation process 24 also transmits an end notification to retransmission request process 25 and reception process 22 (steps 124, 125).

[0070]

When reception process 22 receives the end notification, it resets the timer for obstruction detection (step 104). When retransmission request process 25 receives the end notification, it transmits an end notification to retransmission reply process 15 of transmitting device 10 using TCP/IP (step 151).

[0071]

Having received the end notification, retransmission reply process 15 transmits this end notification to retransmission management process 14 (step 81).

[0072]

In addition, transmission process 12 that transmitted the end text transmits an end notification to retransmission management process 14 (step 36).

[0073]

When retransmission management process 14 receives the end notification from file transmission process 12, it sets an end-waiting timer (step 62). This end-waiting timer is for the purpose of confirming that the end text was correctly received by all of the receiving devices 20, so as will be explained later, the time setting for this counter is set long enough to include the

time required for a retransmission request from receiving devices 20 and the retransmission of the end text in response thereto.

[0074]

Retransmission management process 14 checks whether end notifications have been sent from all of the receiving devices 20 within the time set for this end-waiting timer. If end notifications have been sent from all receiving devices 20, file transmission process 12 is notified that [the transmission] was completed correctly, and the end-waiting timer is reset (step 63). By means of this process, it is possible to ascertain that the file transfer was completed correctly for all of the receiving devices 20 for which the file transfer was intended, and whether there is a faulty receiving device.

[0075]

Figure 6 and Figure 7 show the process when one or more successive texts is not received by one or more receiving devices 20 for some reason.

[0076]

As described above, file transmission process 12 transmits successive texts at a fixed period of time (T1 (steps 37, 38, 39, 40). In the explanation that follows, the number in parentheses that follows the term successive text indicates the serial number. For example, 'successive text (10)' is the successive text having serial number 10. The same is true for the starting text, end text, retransmission requests, and retransmission reply text.

[0077]

When reception process 22 and file creation process 24 of receiving devices 20 receive successive text without an error, the processes in steps 102 and 122 that were explained above are performed.

[0078]

Assume that successive texts (11) and (12) are lost and not delivered to reception process 22.

[0079]

When reception process 22 receives the following successive text (13), in the same manner as in step 102, the received successive text is written to reception data buffer 23 and a reception notification is transmitted to file creation process 24 (step 105).

[0080]

When the reception notification is received, file creation process 24 performs the aforementioned serial number check. The serial number for the text received last is 10 and the serial number for the text received at this time is 13, so file creation process 24 discovers that the text for serial numbers 11 and 12 is missing. Then, the file creation process transmits a first retransmission request regarding serial numbers 11 and 12; that is, it transmits a retransmission request 1 (11, 12) to retransmission request process 25 (step 126).

[0081]

When this retransmission request is received, retransmission request process 25 transmits retransmission request 1 (11, 12) to retransmission reply process 15 of transmitting device 10 (step 152). This retransmission request text, of course, includes a code indicating that it is retransmission request 1.

[0082]

Even after file creation process 24 has transmitted retransmission request 1, file transmission process 12 continues to transmit successive texts at a fixed time interval, so reception process 22 continues to receive these successive texts (continues to write the received text in reception data buffer 23) and continues to provide reception notifications to file creation process 24. File creation process 24 does not perform a process with respect to the successive texts written to reception data buffer 23 even though it receives the reception notifications. A queue, with the received text in the order it was received, is created in reception data buffer 23 (step 126).

[0083]

Retransmission reply process 15, having received retransmission request 1 (11, 12), passes retransmission request 1 (11, 12) to retransmission management process 14 (step 82).

[0084]

Retransmission management process 14 uses retransmission management table 14A to manage the serial numbers of text for which retransmission has been requested. As described previously, in response to retransmission request 1, the text for the serial number for which retransmission was requested is distributed to all of the receiving devices 20 by means of UDP/IP. Accordingly, it is not necessary to retransmit again and again when there are retransmission requests from receiving devices 20 for text with the same serial number.

Therefore, retransmission management process 14 checks to determine whether there already has been a retransmission request for the text with the serial number for which retransmission is being requested, and if [said text] already has been redistributed, that text is not retransmitted.

[0085]

Retransmission management process 14 refers to retransmission management table 14A, realizes that retransmission for this serial number is being requested for the first time, and transmits retransmission request 1 (11, 12) to file transmission process 12 (step 64).

[0086]

When file transmission process 12 receives retransmission request 1, it temporarily stops transmitting successive text, reads from retransmission data buffer 13 the texts for the serial numbers for which retransmission has been requested, and distributes these to retransmission request process 25 as retransmission reply text 1 (11, 12) using UDP/IP (step 41).

[0087]

Retransmission reply process 25 receives this retransmission reply text, stores it in retransmission data buffer 26, and notifies file creation process 24 that a retransmission reply text has been received (step 153).

[0088]

File creation process 24 checks the serial numbers included in the retransmitted text, updates the serial numbers in serial number management table 24A, and stores that text in temporary storage file 27 (step 127). Subsequently, [the process flow] moves to the same type of processing for the successive text waiting in reception data buffer 23 (step 128).

[0089]

Figure 8 and Figure 9 show the process flow for a special case of loss – the loss of staring text. In these figures, the processes that essentially are identical to those in Figure 4 through Figure 7 are denoted with the same symbols.

[0090]

As described previously, file transmission process 12 transmits successive text (1) after starting text (0) (steps 33, 34).

[0091]

Assume that reception process 22 cannot receive starting text (0), but receives the successive text (1) that follows (step 102).

[0092]

File creation process 24 detects by means of the serial number check that starting text (0) is missing, so it transmits a starting text retransmission request 1 (0) and does not process subsequently received text (including successive text (1)) (step 129).

[0093]

Retransmission request 1 (0) is supplied from file creation process 24 to retransmission management process 14 via retransmission request process 25 and retransmission reply process 15 (steps 152, 82).

[0094]

Retransmission process 14 confirms that this is the first retransmission request for the starting text and supplies retransmission request 1 (0) to file transmission process 12 (step 64).

[0095]

File transmission process 12 temporarily stops transmitting successive text and distributes a retransmission reply text (0) for the starting text to retransmission request process 25 using UDP/IP (step 41).

[0096]

Retransmission request process 25 receives this retransmission reply text (0), stores it in retransmission data buffer 26, and notifies file creation process 24 that [text] has been received (step 153).

[0097]

File creation process 24 confirms that this is starting text with a serial number of 0, stores that serial number in serial number management table 24A, and opens a file (step 121). Subsequently, file creation process 24 moves to the processing of the successive text stored in reception data buffer 23.

[0098]

Figure 10 shows the process flow pertaining to a retransmission request 2 that occurs when retransmitted successive text is again lost. This figure is drawn as a process that follows that of Figure 6.

[0099]

As explained with reference to Figure 6 and Figure 7, in reply to retransmission request 1 (11, 12) with respect to the missing successive texts (11) and (12), file transmission process 12 retransmits successive text (11) and successive text (12) (step 41).

[0100]

After retransmission request process 25 of the receiving devices 20 transmits retransmission request 1 (11, 12) (step 152 in Figure 6), [the process] sets a timer for retransmission monitoring use and monitors to determine whether retransmission reply text is received within a prescribed time.

[0101]

As shown in Figure 10, the retransmitted successive text (12) is received, but successive text (11) cannot be received within the aforementioned prescribed time. In a case such as this, retransmission request process 25 transmits a second retransmission request that specifies the same text as the serial numbers (11, 12) specified by retransmission request 1, in other words, it transmits retransmission request 2 (11, 12), to retransmission reply process 15 using TCP/IP (step 154).

[0102]

The retransmitted successive text (12) was received, so a retransmission request 2 for successive text (11) only can be transmitted; however, to avoid complicating the process, the second retransmission request is sent with respect to all of the text that was lost the first time.

[0103]

When retransmission request 2 (11, 12) is received, retransmission reply process 15 reads the requested text (11, 12) from retransmission data buffer 13 and transmits it to retransmission request process 25 as retransmission reply text 2 (11, 12) by means of TCP/IP (step 83). Thus, data are retransmitted reliably for the second retransmission.

[0104]

When retransmission reply text 2 (11, 12) is received, retransmission request process 25 stores the received text in retransmission data buffer 26 and notifies file creation process 24 that text has been received (step 155).

[0105]

When this notification is received, file creation process 24 checks and updates the serial numbers and stores the retransmitted text in temporary storage file 27 (step 127). Subsequently, processing of the successive text held in reception data buffer 23 proceeds (step 128).

[0106]

Figure 11 shows the flow for a second retransmission for starting text. This figure is drawn as a process that follows that of Figure 8. The steps that are identical to those of the processes shown in Figure 9 and the previously described Figure 10 are denoted with the same symbols, so the process for this case can be understood easily from Figure 11.

[0107]

Figure 12 shows the flow of a mandatory end process due to a reception data buffer overflow. This figure is drawn as a process that follows that of Figure 6.

[0108]

A retransmission request 1 (11, 12) is generated due to the loss of successive text (11, 12), and in reply thereto file transmission process 12 redistributes successive text (11, 12) (step 41). Subsequently, file transmission process 12 restarts distribution beginning with successive text (14) (steps 40, 42).

[0109]

Reception process 22 receives this successive text (14) and the like and stores the text sequentially in reception data buffer 23 (step 105). File creation process 24 is in the idle state until processing of the retransmitted text, so it does not read text from reception data buffer 23 (step 130).

[0110]

When there is a large amount of retransmitted text, and if that is not delivered to retransmission request process 25, as time passes, a large amount of text will accumulate in reception data buffer 23, resulting in an overflow.

[0111]

There is a flag for each area where text is stored in reception data buffer 23; when reception process 22 writes text, these flags are erected, and when file creation process 24 reads text, these flags is reset. Accordingly, based on the status of these flags, reception process 22 can determine whether there is no more room in reception data buffer 23 for writing of text.

[0112]

When reception data buffer 23 is full and no more text can be written, reception process 22 transmits to file creation process 24 [a message] indicating a mandatory stop (step 106). When this is received, file creation process 24 transmits a mandatory stop to retransmission request process 25 (step 131).

[0113]

When this mandatory stop is received, retransmission request process 25 resets the aforementioned retransmission monitor timer and transmits [a message] indicating the mandatory stop to retransmission reply process 15 by means of TCP/IP (step 156).

[0114]

Retransmission reply process 15 passes the received mandatory stop to retransmission management process 14 (step 84), so retransmission management process 14 stores in memory [information to the effect] that file transfer has been mandatorily stopped for that receiving device, and notifies file transmission process 12 to that effect (step 65). Of course, file transmission process 12 continues to distribute successive text to the other receiving devices 20 that are normal.

[0115]

Figure 13 and Figure 14 show the process flow when a transmission path obstruction occurs and is subsequently resolved. The steps that are identical to those of the processes in Figure 6 and Figure 7 are denoted with identical symbols.

[0116]

As described above, reception process 22 is equipped with a timer for detection of obstructions, and thus monitors to determine whether there has been no communication for longer than the fixed time period T2 that is longer than the distribution period T1.

[0117]

At the reception process 22 of a receiving device for which text has not been received for longer than the aforementioned fixed time period T2 due to a transmission path obstruction, the obstruction detection timer times out, so that reception process 22 transmits an obstruction notification to file creation process 24 (step 107).

[0118]

When this notification is received, file creation process 24 appends the serial number of the last correctly received text and transmits the obstruction notification to retransmission request process 25 (step 131). Retransmission request process 25 transmits this obstruction notification to retransmission reply process 15 using TCP/IP (step 157).

[0119]

This obstruction notification is passes from retransmission reply process 15 to retransmission management process 14 (step 85). When retransmission management process 14 receives the obstruction notification, it stores in retransmission management table 14A the identification number of the receiving device 20 that transmitted the obstruction notification and the serial number of the text last received by that receiving device, and supplies the obstruction notification to file transmission process 12 (step 66).

[0120]

When the obstruction notification is received, file transmission process 12 distributes dummy text to all of the receiving devices 20 at the aforementioned fixed time period T1 (step 43).

[0121]

When a receiving device that did not generate the obstruction notification receives the dummy text, it recognizes that an obstruction has occurred with one of the receiving devices.

[0122]

As long as the obstruction continues, the receiving device that generated the obstruction notification does not receive dummy text. When reception process 22 of the receiving device 20 that generated the obstruction notification receives dummy text, it means that the obstruction has been resolved. Accordingly, when that reception process 22 receives dummy text, it transmits a recovery notification to file creation process 24 (step 108). The recovery notification is

transmitted from retransmission request process 25 via file creation process 24 to retransmission reply process 15 of transmitting device 10 by means of TCP/IP (steps 133, 158).

[0123]

The recovery notification then is supplied from retransmission reply process 15 to retransmission management process 14 (step 86).

[0124]

As described above, retransmission management process 14 stores in retransmission management table 14A the receiving device 20 for which the obstruction occurred and the serial number of the last text received by that receiving device. If obstructions have occurred at multiple receiving devices 20, retransmission management process 14 waits until recovery notifications from all of those receiving devices are received.

[0125]

If, within an obstruction recovery monitor time to be explained later, recovery notifications are transmitted from all of the receiving devices 20 for which an obstruction occurred, retransmission management process 14 supplies a recovery notification to file transmission process 12, and notifies file transmission process 12 of the smallest serial number among the last serial numbers received by the receiving devices for which an obstruction occurred (step 67).

[0126]

When this serial number is received, file transmission process 12 restarts the distribution of the successive text beginning with the serial number that is one greater than the aforementioned smallest serial number (step 44). Thus, when an obstruction occurs and is resolved, the successive text is distributed to the receiving devices 20 without omitting a number (there may be duplication [of text] for some of the receiving devices).

[0127]

When the distributed text is received by the receiving devices 20, the processes corresponding thereto are restarted (steps 102, 122).

[0128]

Figure 15 shows the process flow when a transmission path obstruction occurs and is not resolved. This picture is drawn as a process that follows that in Figure 13.

[0129]

When retransmission management process 14 of transmitting device 10 receives the first obstruction notice, it activates the aforementioned obstruction recovery monitor timer. Even if the obstruction recovery monitor timer has timed out, if a recovery notification has not been transmitted from one or more of the receiving devices 20 for which an obstruction occurred, retransmission management process 14 supplies to retransmission reply process 15 [a message] indicating a mandatory stop in order to mandatorily stop the receiving device 20 that has not transmitted a recovery notification. In addition, if there is even one receiving device for which an obstruction has not occurred or a receiving device that has recovered from an obstruction, the recovery notification is supplied to file transmission process 12 (step 68).

[0130]

When the recovery notification is received, as described above, file transmission process 12 restarts the distribution of the successive text (step 44).

[0131]

When retransmission reply process 15 receives the notification indicating a mandatory stop, it uses TCP/IP to transmit to retransmission request process 25 of the receiving device 20 that has not recovered from the obstruction [a message] indicating the mandatory stop (step 87).

[0132]

If the retransmission reply process 25 in question receives this mandatory stop notification, this mandatory stop notification is reported to reception process 22 via file creation process 24. Accordingly, even if reception process 22 subsequently receives text, it is discarded (steps 159, 134, 109).

[0133]

Figure 16 shows the process flow when text with the same serial number is received again.

[0134]

As described above, when a receiving device 20 for which an obstruction occurs has recovered, text with the same serial number may be received. File creation process 24 performs the aforementioned serial number check, so it is possible to discover that there are duplicate

serial numbers. In this case, file creation process 24 discards the text that is received again without writing it to temporary storage file 27 (step 135).

[0135]

Figure 17 and Figure 18 show the process flow when the last text is lost. Basically, this case is identical to the process for a retransmission request 1. Therefore, the steps that essentially are identical to those shown in Figure 4, Figure 5, Figure 6, and Figure 7 are denoted with the same symbols.

[0136]

Rather than with a serial number check, loss of the last text is detected in the same manner that an obstruction is detected, when no communication has occurred for longer than the fixed time T2; therefore, reception process 22 transmits an obstruction notification to file creation process 24 (step 110).

[0137]

The serial number of the last text is already known based on the content of the data (main body) of the starting text, so if the text up to the serial number that is one before that has been received normally, file creation process 24 recognizes that an error has occurred with the last text and transmits a retransmission request 1(100) to retransmission request process 25 (step 136).

[0138]

Retransmission request process 25 transmits retransmission request 1(100) to retransmission reply process 15 (step 152), so the last text is redistributed from file transmission process 12 (steps 82, 64, 46).

[0139]

If this last text can be received correctly by the receiving device, an end notification is transmitted from receiving device 20 to transmitting device 10, and the file transfer is finished (steps 153, 151, 123, 125, 104, 81, 63, 47). After file transmission process 12 supplies the end notification to retransmission management process 14 (step 36), if there is a receiving device that has not transmitted an end notification after the set time for the end-waiting timer has elapsed, retransmission management process 14 records a file transfer failure with respect to that receiving device.

[0140]

Of course, not only a first but a second retransmission can be performed for a loss of the last text.

Brief description of the figures

Figure 1 is a block diagram showing the overall configuration of a multiple address file transfer system.

Figure 2 is a block diagram showing the functional configuration of a transmitting device and a receiving device.

Figure 3 shows the file transfer text format.

Figure 4 is a flow chart showing the process flow when a file transfer occurs correctly.

Figure 5 is a flow chart showing the process flow when a file transfer occurs correctly.

Figure 6 is a flow chart showing the flow of the first retransmission process due to missing successive electronic text.

Figure 7 is a flow chart showing the flow of the first retransmission process due to missing successive electronic text.

Figure 8 is a flow chart showing the flow of the first retransmission process due to missing initial electronic text.

Figure 9 is a flow chart showing the flow of first retransmission process due to missing initial electronic text.

Figure 10 is a flow chart showing the flow of the second retransmission process due to missing successive electronic text.

Figure 11 is a flow chart showing the flow of the second retransmission process due to missing initial electronic text.

Figure 12 is a flow chart showing the flow of the end process due to a reception data buffer overflow.

Figure 13 is a flow chart showing the process flow when an obstruction occurs in the transmission path and it is resolved subsequently.

Figure 14 is a flow chart showing the process flow when an obstruction occurs in the transmission path and it is resolved subsequently.

Figure 15 is a flow chart showing the flow of the end process when an obstruction occurs in the transmission path and recovery is not possible.

Figure 16 is a flow chart showing the process flow when text with the same serial number is received twice.

Figure 17 is a flow chart showing the flow of the first retransmission process due to missing final electronic text.

Figure 18 is a flow chart showing the flow of the first retransmission process due to missing final electronic text.

Explanation of symbols

8	Transmission path
9	Relay device
10	Transmitting device
11	Transmission file
12	File transmission process
13	Retransmission data buffer
14	Retransmission management process
14A	Retransmission table
15	Retransmission reply process
20	Receiving device
21	Reception file
22	Reception process
23	Reception data buffer
24	File creation process
24A	Serial number management table
25	Retransmission request process
26	Retransmission data buffer
27	Temporary storage file

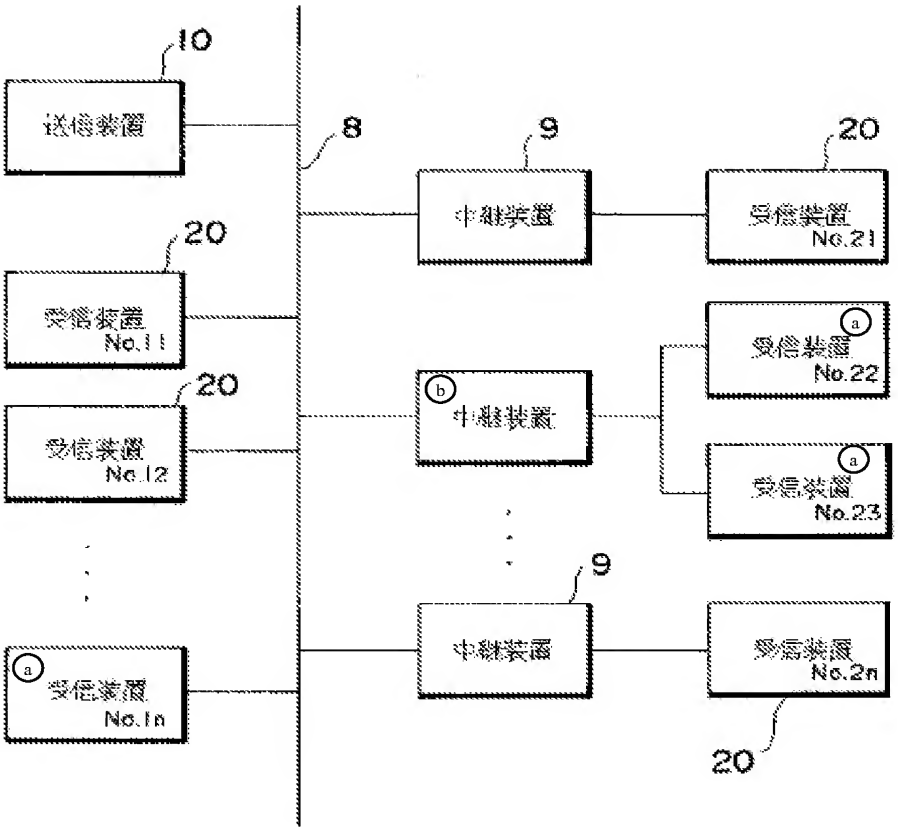


Figure 1

Key:
- a Receiving device
- b Relay device
- 9 Relay device
- 10 Transmitting device
- 20 Receiving device

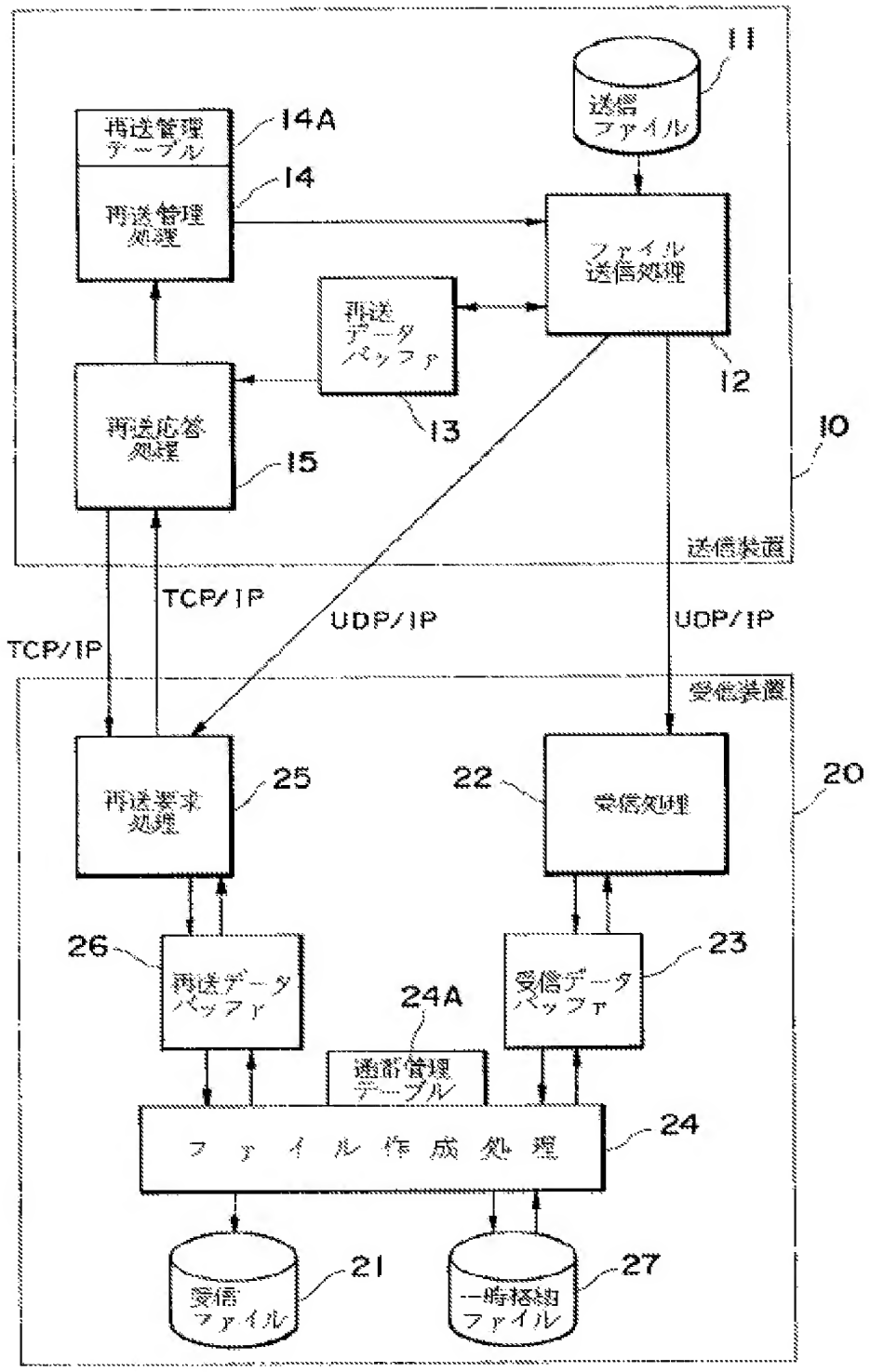

11
送信ファイル
14A
再送管理テーブル
14
再送管理処理
ファイル送信処理
再送データバッファ
13
12
再送応答処理
15
10
送信装置
TCP/IP
TCP/IP
UDP/IP
UDP/IP
受信装置
再送要求処理
25
22
受信処理
20
26
再送データバッファ
24A
通信管理テーブル
受信データバッファ
23
ファイル作成処理
24
受信ファイル
21
一時格納ファイル
27

Figure 2

Key:
10 Transmitting device
11 Transmission file
12 File transmission process
13 Retransmission data buffer
14 Retransmission management process
14A Retransmission table
15 Retransmission reply process
20 Receiving device
21 Reception file
22 Reception process
23 Reception data buffer
24 File creation process
24A Serial number management table
25 Retransmission request process
26 Retransmission data buffer
27 Temporary storage file

Text / Item	Start	Successive	End	Dummy
Text class	FT	FT	FT	FT
Resource ID	FT	FT	FT	FT
Text class details	ST	MD	LS	DM
File transfer ID	ID appended per file transfer	Used with 'start'	Used with 'start'	Used with 'start'
Text serial number	0	Sequentially from 1	Last serial number	
Record length	Data length	Data length		
Data	File name; file size; last serial number; execution shell script	File data	File data	

Figure 3

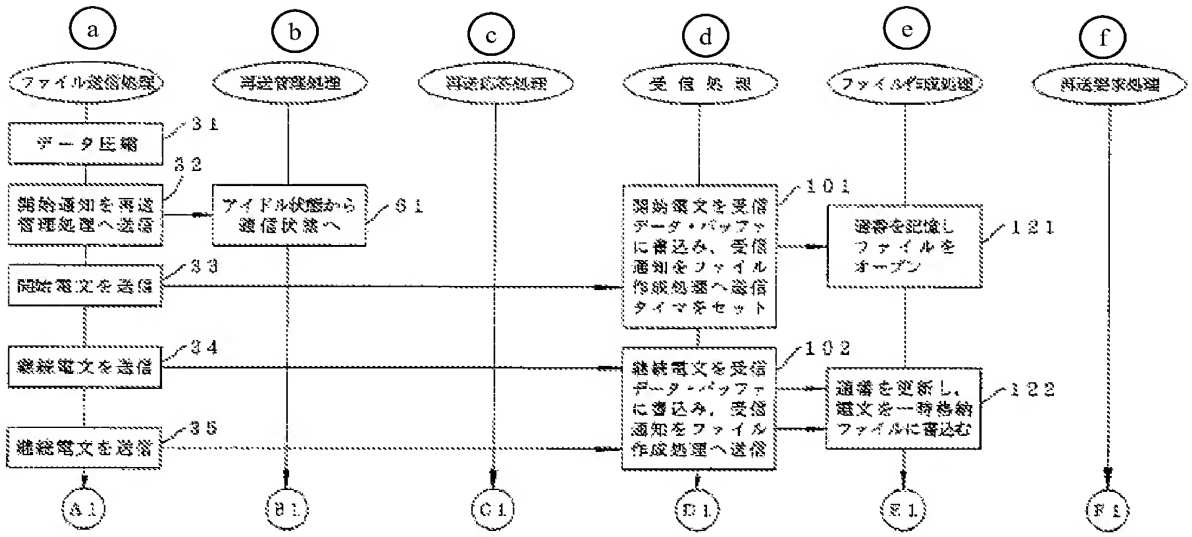


Figure 4

Key:

a	File transmission process
b	Retransmission management process
c	Retransmission reply process
d	Reception process
e	File creation process
f	Retransmission request process
31	Data compression
32	Transmit starting serial number to retransmission management process
33	Transmit starting text
34	Transmit successive text
35	Transmit successive text
61	Move from idle state to communication state
101	Write starting text to reception data buffer, transmit reception notification to file creation process, set timer
102	Write successive text to reception data buffer, transmit reception notification to file creation process
121	Store serial number and open file
122	Update serial number, write text to temporary storage file

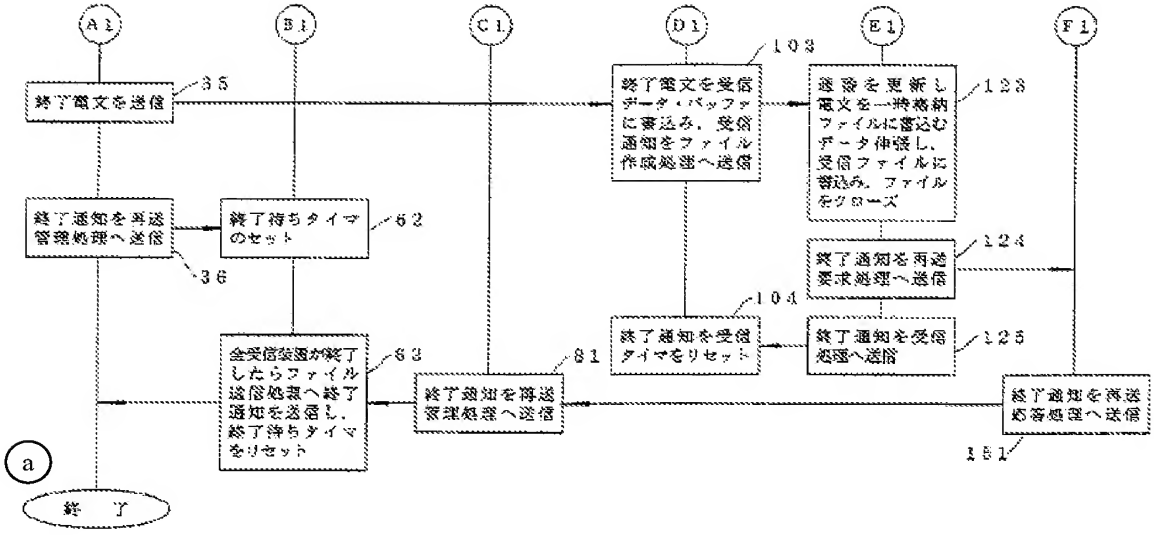


Figure 5

Key:

a	End
35	Transmit end text
36	Transmit end notification to retransmission management process
62	Set end-waiting timer
63	When all receiving devices have finished, transmit end notification to file transmission process and reset end-waiting timer
81	Transmit end notification to retransmission management process
103	Write last text to reception data buffer, transmit reception notification to file creation process
104	Receive end notification, reset timer
123	Update serial number, write text to temporary storage file, expand data, write to reception file, close file
124	Transmit end notification to retransmission request process
125	Transmit end notification to reception process
151	Transmit end notification to retransmission reply process

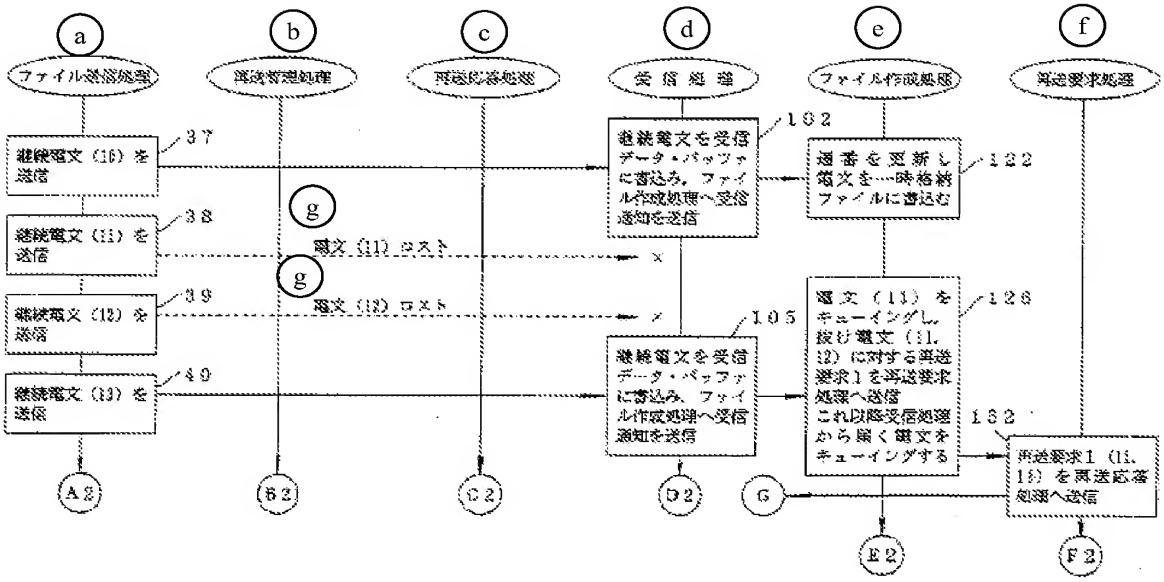


Figure 6

Key:
- a File transmission process
- b Retransmission management process
- c Retransmission reply process
- d Reception process
- e File creation process
- f Retransmission request process
- g Text ___ lost
- 37 Transmit successive text (10)
- 38 Transmit successive text (11)
- 39 Transmit successive text (12)
- 40 Transmit successive text (13)
- 102 Write successive text to reception data buffer, transmit reception notification to file creation process
- 105 Write successive text to reception data buffer, transmit reception notification to file creation process
- 122 Update serial number, write text to temporary storage file
- 126 Queue text (13), transmit retransmission request 1 for missing texts (11, 12) to retransmission request process, queue any text subsequently delivered from the reception process
- 152 Transmit retransmission request 1 (11, 12) to retransmission reply process

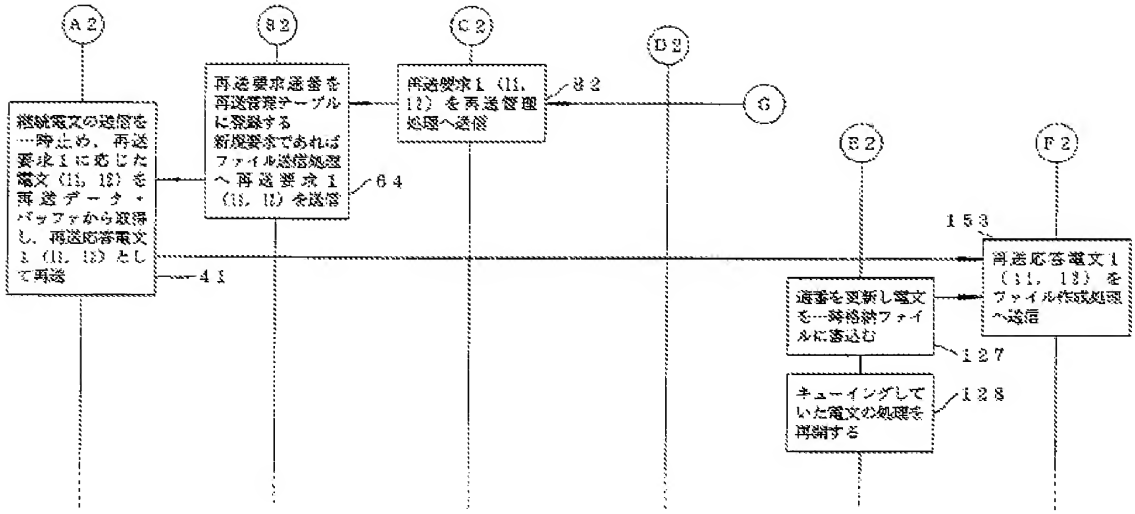


Figure 7

Key: 41 Temporarily stop transmission of successive text, acquire text (11, 12) from retransmission data buffer according to retransmission request 1, retransmit as retransmission reply text 1(11, 12)

64 Register retransmission request serial number in retransmission management table; if this is a new request, transmit retransmission request 1(11, 12) to file transmission process

82 Transmit retransmission request 1(11, 12) to retransmission management process

127 Update serial number, write text to temporary storage file

128 Restart process for queued text

153 Transmit retransmission reply text 1(11, 12) to file creation process

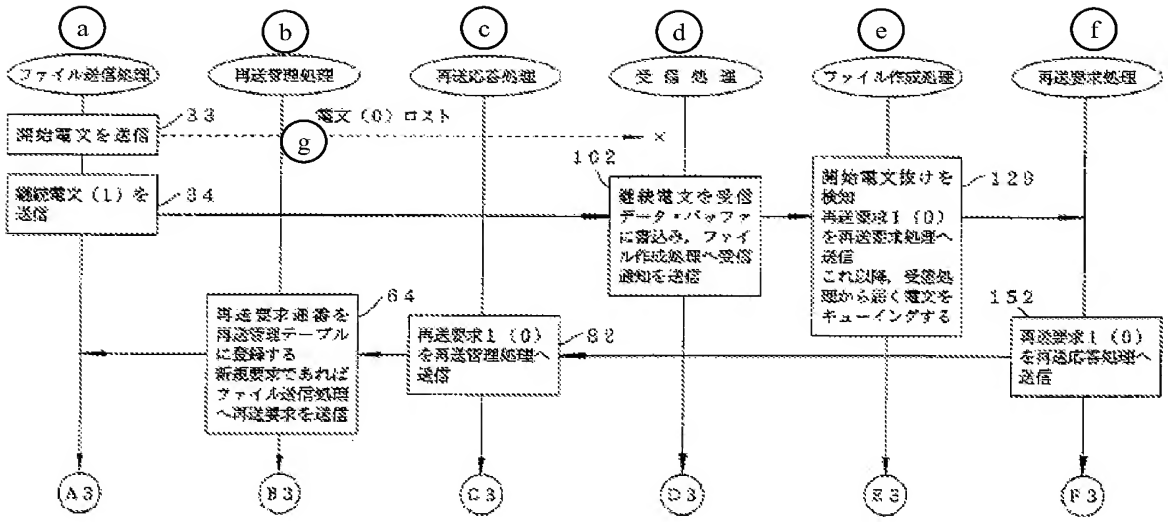


Figure 8

Key:
- a File transmission process
- b Retransmission management process
- c Retransmission reply process
- d Reception process
- e File creation process
- f Retransmission request process
- g Text (0) lost
- 33 Transmit starting text
- 34 Transmit successive text
- 64 Register retransmission request serial number in retransmission management table; if this is a new request, transmit retransmission request to file transmission process
- 82 Transmit retransmission request 1(0) to retransmission management process
- 102 Write successive text to reception data buffer, transmit reception notification to file creation process
- 129 Detect missing starting text, transmit retransmission request 1(0) to retransmission request process, queue any text subsequently delivered from the reception process
- 152 Transmit retransmission request 1(0) to retransmission reply process

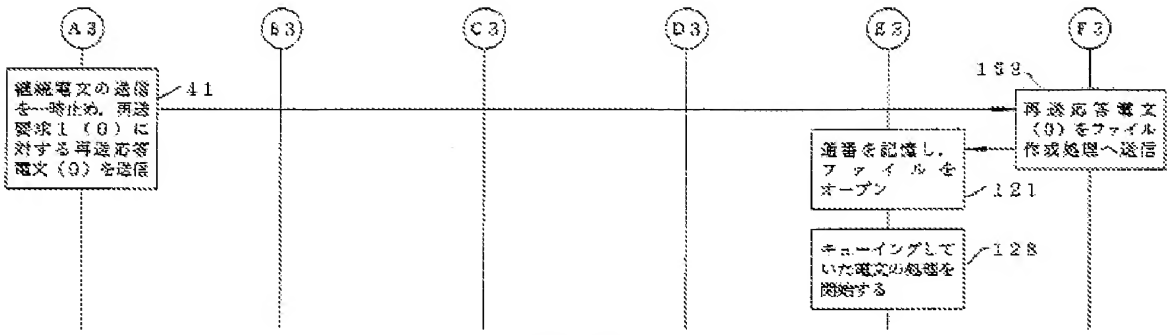


Figure 9

Key: 41 Temporarily stop transmission of successive text, transmit retransmission reply text (0) with respect to retransmission request 1(0)
121 Store serial number, open file
128 Start process for queued text
153 Transmit retransmission reply text (0) to file creation process

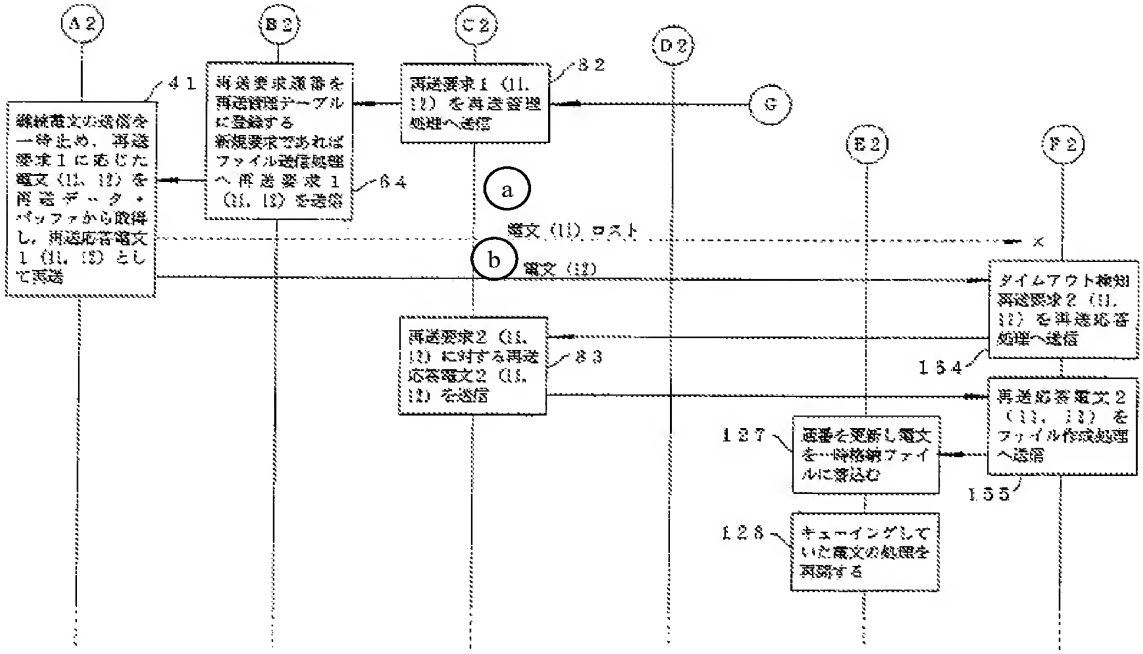


Figure 10

Key: a Text (11) lost
b Text (12)
41 Temporarily stop transmission of successive text, acquire text (11, 12) from retransmission data buffer according to retransmission request 1, retransmit as retransmission reply text 1(11, 12)
64 Register retransmission request serial number in retransmission management table; if this is a new request, transmit retransmission request 1(11, 12) to file transmission process
82 Transmit retransmission request 1(11, 12) to retransmission management process
83 Transmit retransmission reply text 2(11, 12) for retransmission request 2(11, 12)
127 Update serial number, write text to temporary storage file
128 Restart process for queued text
154 Detect timeout, transmit retransmission request 2(11, 12) to retransmission reply process
155 Transmit retransmission reply text 2(11, 12) to file creation process

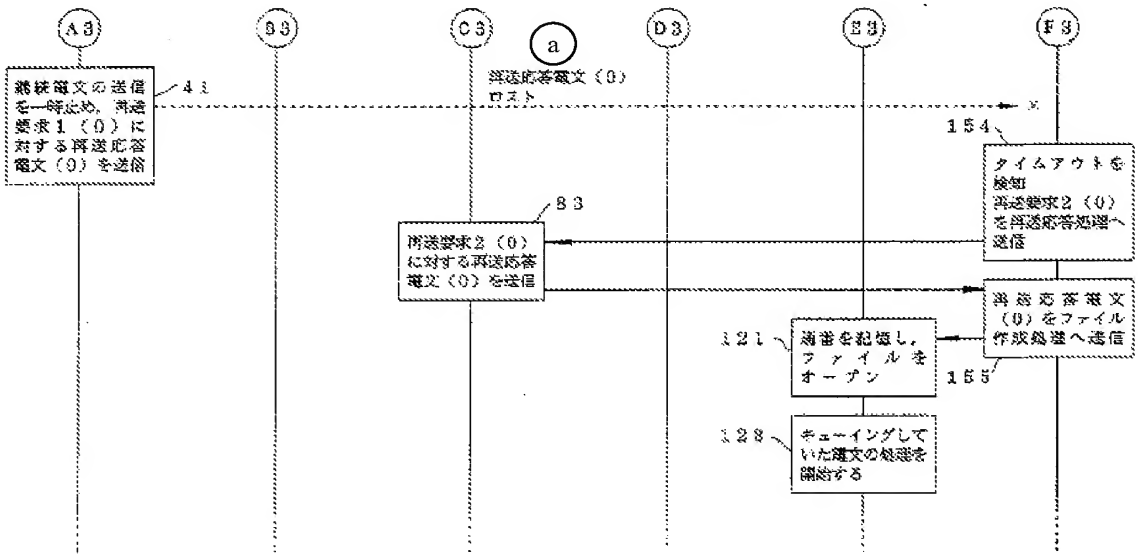


Figure 11

Key: a Retransmission reply text (0) lost
41 Temporarily stop transmission of successive text, transmit retransmission reply text (0) with respect to retransmission request 1(0)
83 Transmit retransmission reply text (0) for retransmission request 2(0)
121 Store serial number and open file
128 Start process for queued text
154 Detect timeout, transmit retransmission request 2(0) to retransmission reply process
155 Transmit retransmission reply text (0) to file creation process

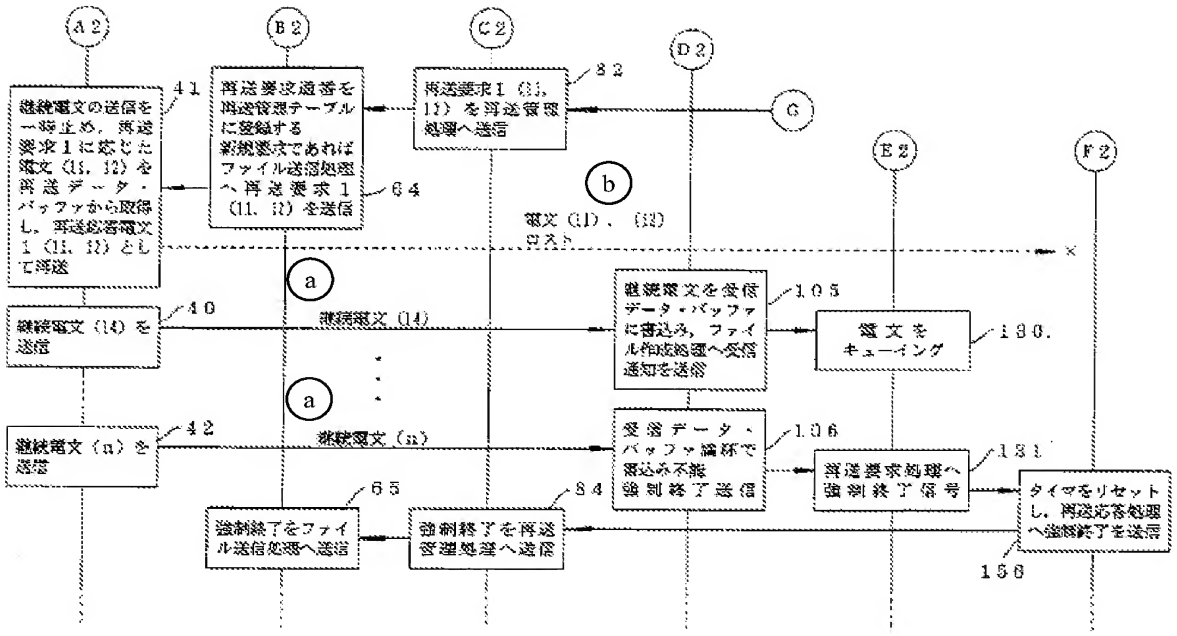


Figure 12

Key:
- a Successive text __
- b Text (11), (12) lost
- 40 Transmit successive text (14)
- 41 Temporarily stop transmission of successive text, acquire text (11, 12) from retransmission data buffer according to retransmission request 1, retransmit as retransmission reply text 1(11, 12)
- 42 Transmit successive text (n)
- 64 Register retransmission request serial number in retransmission management table; if this is a new request, transmit retransmission request 1(11, 12) to file transmission process
- 65 Transmit mandatory stop to file transmission process
- 82 Transmit retransmission request 1(11, 12) to retransmission management process
- 84 Transmit mandatory stop to retransmission management process
- 105 Write successive text to reception data buffer, transmit reception notification to file creation process
- 106 Reception data buffer is full, so writing is not possible; transmit mandatory stop
- 130 Queue text
- 131 Mandatory stop signal [sic] to retransmission request process
- 156 Reset timer, transmit mandatory stop to retransmission reply process

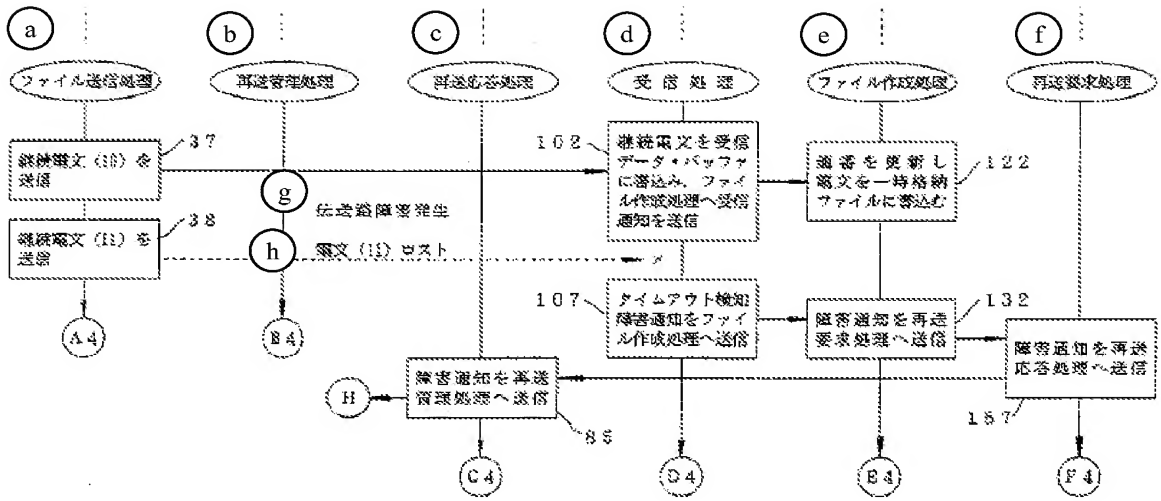


Figure 13

Key:

a	File transmission process
b	Retransmission management process
c	Retransmission reply process
d	Reception process
e	File creation process
f	Retransmission request process
g	Transmission path obstruction occurs
h	Text (11) lost
37	Transmit successive text (10)
38	Transmit successive text (11)
85	Transmit obstruction notification to retransmission management process
102	Write successive text to reception data buffer, transmit reception notification to file creation process
107	Detect timeout, transmit obstruction notification to file creation process
122	Update serial number, write text to temporary storage file
132	Transmit obstruction notification to retransmission request process
157	Transmit obstruction notification to retransmission reply process

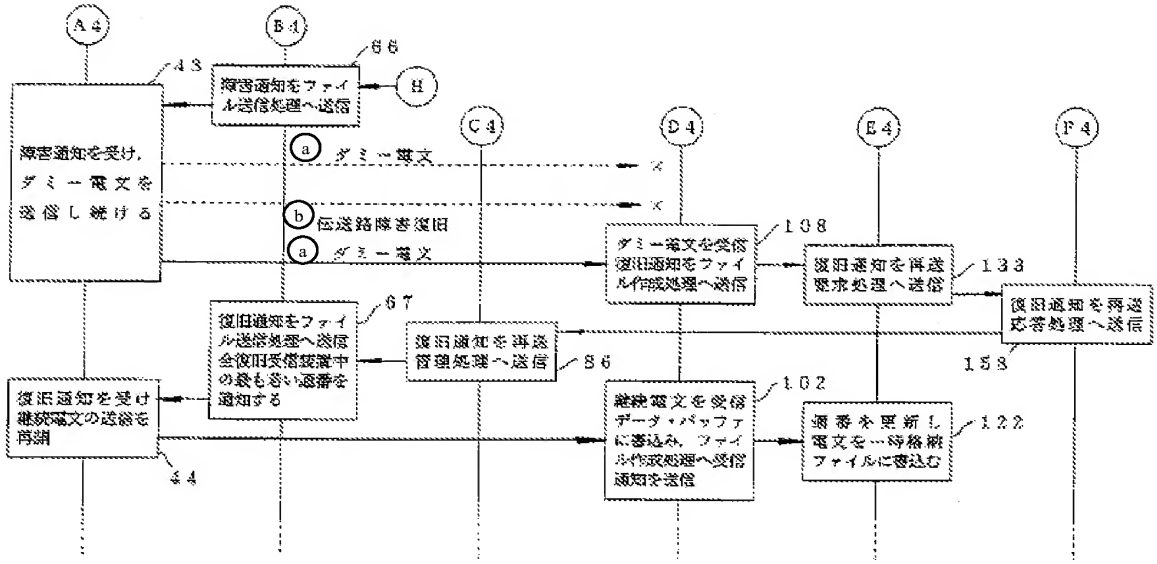


Figure 14

Key:
- a Dummy text
- b Transmission path obstruction resolved
- 43 Receive obstruction notification, continue transmitting dummy text
- 44 Receive recovery notification, restart transmission of successive text
- 66 Transmit obstruction notification to file transmission process
- 67 Transmit recovery notification to file transmission process, provide notice of smallest serial number among all of the receiving devices that have recovered
- 86 Transmit recovery notification to retransmission management process
- 102 Write successive text to reception data buffer, transmit reception notification to file creation process
- 108 Receive dummy text, transmit recovery notification to file creation process
- 122 Update serial number, write text to temporary storage file
- 133 Transmit recovery notification to retransmission request process
- 158 Transmit recovery notification to retransmission reply process

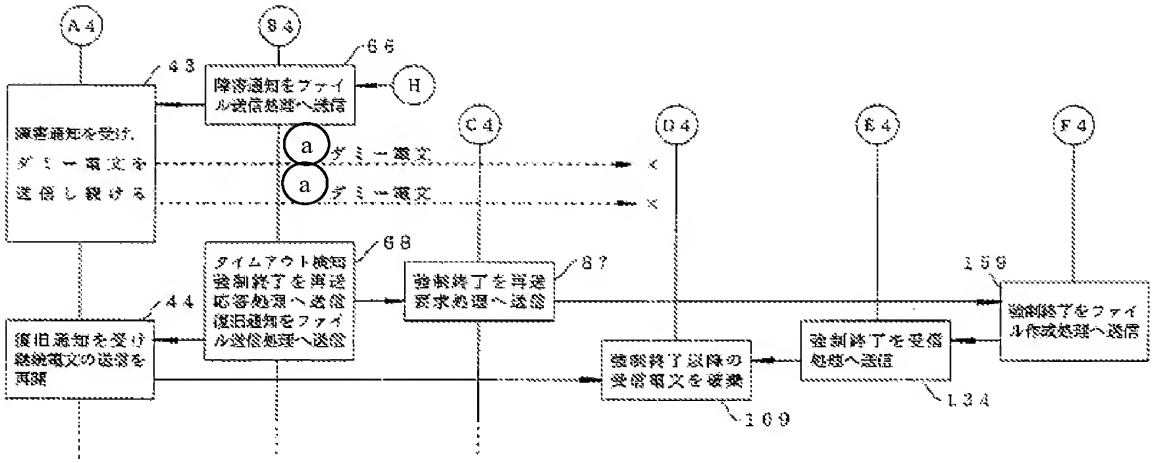


Figure 15

Key:
- a — Dummy text
- 43 — Receive obstruction notification, continue transmitting dummy text
- 44 — Receive recovery notification, restart transmission of successive text
- 66 — Transmit obstruction notification to file transmission process
- 68 — Detect timeout, transmit mandatory stop to retransmission reply process, transmit recovery notification to file transmission process
- 87 — Transmit mandatory stop to retransmission request process
- 109 — Discard any text subsequent to mandatory stop
- 134 — Transmit mandatory stop to reception process
- 159 — Transmit mandatory stop to file creation process

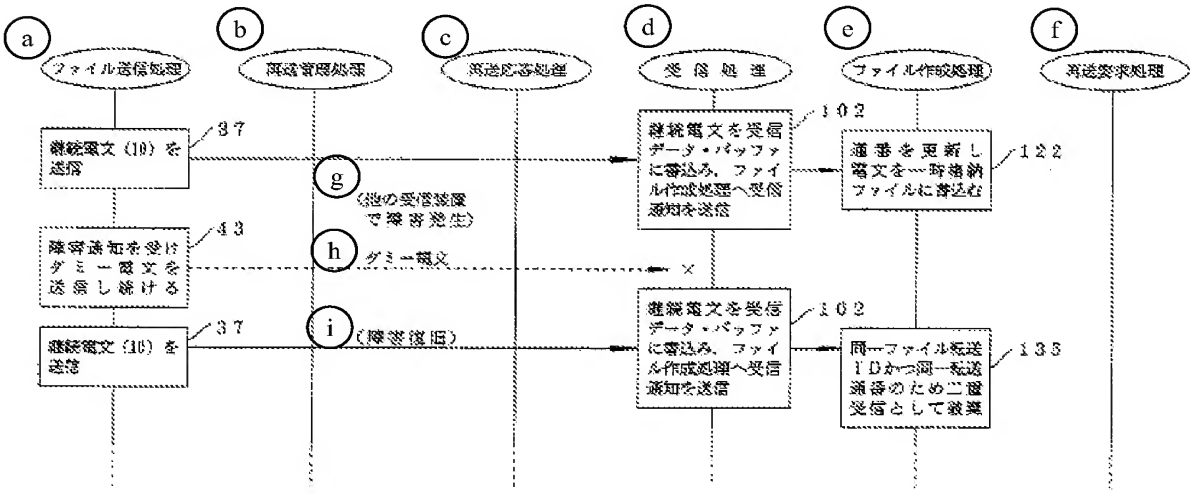


Figure 16

Key:
- a File transmission process
- b Retransmission management process
- c Retransmission reply process
- d Reception process
- e File creation process
- f Retransmission request process
- g (obstruction occurs with another receiving device)
- h Dummy text
- i (obstruction resolved)
- 37 Transmit successive text (10)
- 43 Receive obstruction notification, continue transmitting dummy text
- 102 Write successive text to reception data buffer, transmit reception notification to file creation process
- 122 Update serial number, write text to temporary storage file
- 135 File transfer ID is identical and transfer serial number is identical, so discard as a redundant reception

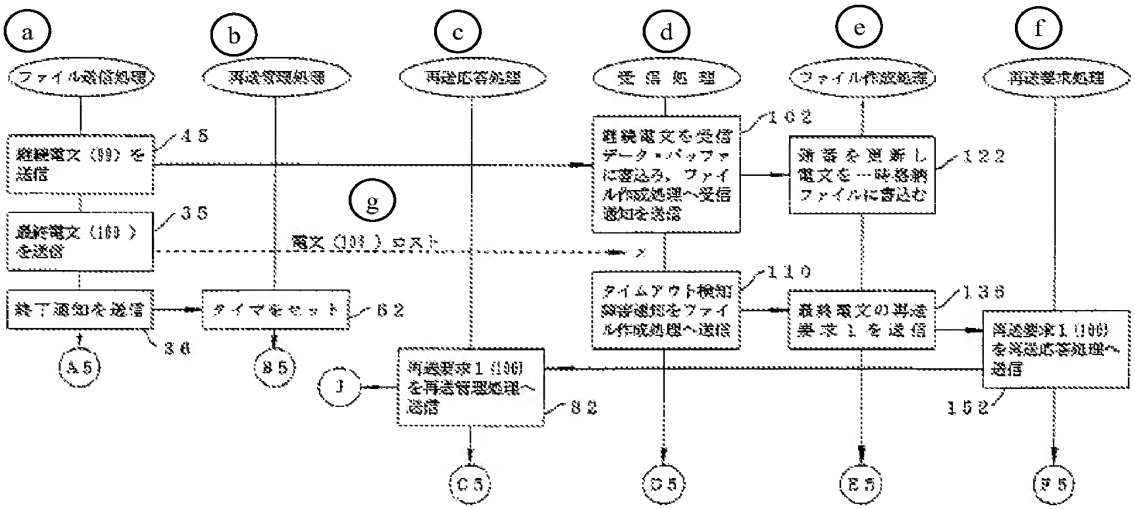


Figure 17

Key:
- a File transmission process
- b Retransmission management process
- c Retransmission reply process
- d Reception process
- e File creation process
- f Retransmission request process
- g Text (100) lost
- 35 Transmit last text (100)
- 36 Transmit end notification
- 45 Transmit successive text (99)
- 62 Set timer

82 Transmit retransmission request 1(100) to retransmission management process
102 Write successive text to reception data buffer, transmit reception notification to file creation process
110 Detect timeout, transmit obstruction notification to file creation process
122 Update serial number, write text to temporary storage file
136 Transmit retransmission request 1 for last text
152 Transmit retransmission request 1(100) to retransmission reply process

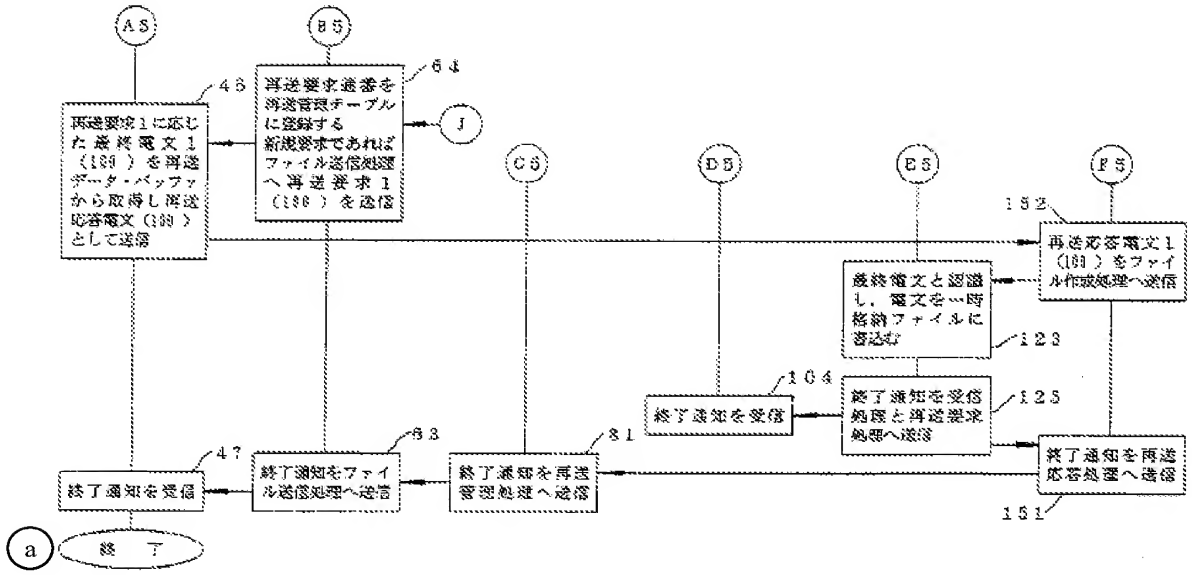


Figure 18

Key: a End
46 Acquire last text 1(100) according to retransmission request 1 from retransmission data buffer and transmit as retransmission reply text (100)
47 Receive end notification
63 Transmit end notification to file transmission process
64 Register retransmission request serial number in retransmission management table; if this is a new request, transmit retransmission request 1(100) to file transmission process
81 Transmit end notification to retransmission management process
104 Receive end notification
123 Confirm as last text, write text to temporary storage file
125 Transmit end notification to reception process and retransmission request process
151 Transmit end notification to retransmission reply process
152 Transmit retransmission reply text 1(100) to file creation process

Continued from front page

(72) Inventor: Kazuhiko Yoda
Nomura Research Institute, Ltd.
9-1-1 Nihonbashi
Chuo-ku, Tokyo

(72) Inventor: Satoru Nishino
Nomura Research Institute, Ltd.
Yokohama Research Center
134 Kobecho, Hodogaya-ku
Yokohama-shi, Kanagawa-ken

(72) Inventor: Tsutomu Miyawaki
Nomura Research Institute, Ltd.
Daini Yamagata Biru,
Nihonbashi Software Center
6-7, Nihonbashi Koamicho
Chuo-ku, Tokyo